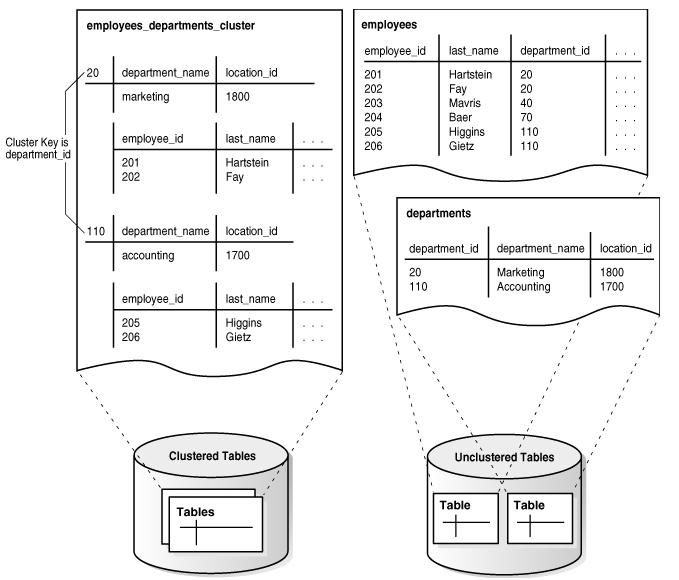
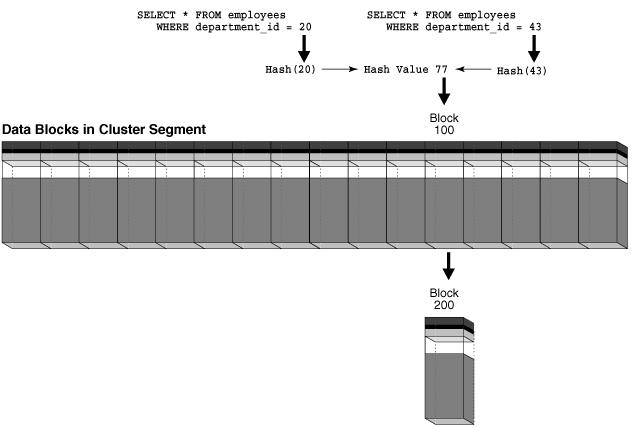
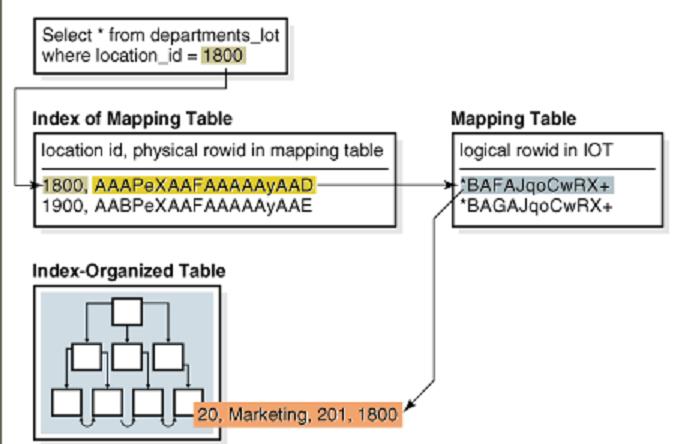
**oracle数据库概念**

**01 概念**   
  
术语   
SGA: system global area   
PGA: program global area   
  
关系数据库的特点：   
1 良好定义的结构 2 清晰定义的操作 3 完整性检查   
  
逻辑操作和物理操作。   
  
Schema Object 逻辑数据结构，比如表，索引。   
  
物理存储结构：   
Data files: 存储数据，table，index。   
Control files: 数据库物理结构的元数据。   
Online redo log files: 记录所有对数据的改动。   
  
逻辑存储结构：   
Data blocks, Extents, Segments, Tablespaces.   
  
数据库 数据库实例   
  
client process | background process | server process 

**02 schema和表**   
  
Each user owns a single schema, which has the same name as the user. The schema contains the data for the user owning the schema.   
  
一个用户有一个schema，一个schema包含多个schema object。   
  
Schema object: Table,Indexes,Partitions,Views,Sequences,Dimensions,Synonyms,PL/SQL subprograms and packages   
  
schema object的依赖。   
当被依赖的schema object改变的时候，需要重新编译依赖它的schema object。   
  
系统自带的SYSTEM schema用来维护整个db，任何人都不应该手动改动它。   
  
样例schema。   
  
表可以分为Relational tables 和 Object tables.   
也可以分为heap-organized table | index-organized table | external table   
也可以分为permanent or temporary   
  
计算列：不占用空间。   
NOT NULL or PK可以保证存储值不为null。   
  
列的类型：   
  
VARCHAR2 and CHAR   
定长字符串VARCHAR2和可变字符串CHAR，Oracle对定长字符串比较的时候会加上padding的空白，注意存储长度的区别。   
NCHAR and NVARCHAR2   
unicode编码的字符串。   
  
Numeric 十进制的数字存储。   
NUMBER   
BINARY\_FLOAT and BINARY\_DOUBLE 2进制的存储方式存储，注意不能精确的存储原值。   
  
Date: 精确到秒 8个byte   
TIMESTAMP: 更精确的时间。可以关联时区。   
  
RowId:   
Physical rowids store the addresses of rows in heap-organized tables, table clusters, and table and index partitions.   
  
Logical rowids store the addresses of rows in index-organized tables.   
  
Format Models and Data Types：指定格式转换，不影响底层真正的存储。   
  
Object Tables   
用户自定义类型，然后用该类型定义表的列。   
  
Temporary Tables   
临时表，Unlike temporary tables in some other relational databases, when you create a temporary table in an Oracle database, you create a static table definition. The temporary table is a persistent object described in the data dictionary, but appears empty until your session inserts data into the table. You create a temporary table for the database itself, not for every PL/SQL stored procedure.   
  
External Tables   
外部表可以把外部资源（比如文件）整合的像是在数据库中一样。   
  
表存储   
默认情况，表记录是无序存储的，即heap-organized，db会自己探测哪里适合存储记录，这就导致了记录是无序的。   
表压缩技术，取出重复冗余数据。   
存储null：尽量节省空间，如果碰到下一行的开始，则这一行剩余的列都是null。   
  
Table Clusters   
一组表共享一些列，并且相关的数据存储在同一个block里。   
   
Indexed Clusters   
就是在Table Cluster上加的索引。   
Hash Clusters   
不用单独的索引结构，用数据本身的hash作为索引。   
  
 

**03 Index**   
  
使用索引可以大大提高数据库的性能。   
  
Primary and unique keys automatically have indexes.   
  
复合索引的列顺序问题。组合列中在前面的列在where子句中查询才会走索引。   
  
Reverse key indexes 防止相近的key聚集。但是没有了Index Range Scan功能。   
  
B-Tree Indexes 典型的B-Tree索引。   
  
Full Index Scan：可以消除排序，因为index本身已经排序。   
  
Suppose that an application runs the following query:SELECT department\_id, last\_name, salary FROM   employeesWHERE  salary > 5000 ORDER BY department\_id, last\_name;Also assume that department\_id, last\_name, and salary are a composite key in an index. Oracle Database performs a full scan of the index, reading it in sorted order (ordered by department ID and last name) and filtering on the salary attribute. In this way, the database scans a set of data smaller than the employees table, which contains more columns than are included in the query, and avoids sorting the data.

Fast Full Index Scan：只扫描Index table，因为Index table已经包含了所有要返回的数据。   
A fast full index scan is a full index scan in which the database reads the index blocks in no particular order. The database accesses the data in the index itself, without accessing the table.   
  
Index Range Scan。   
Index Unique Scan。   
Index Skip Scan：customers table的cust\_gender列不是M就是F,该表有一个复合索引(cust\_gender, cust\_email).   
那么以下这个查询   
SELECT \* FROM sh.customers WHERE cust\_email = 'Abbey@company.com';   
数据库自己优化为：   
SELECT \* FROM sh.customers WHERE cust\_gender = 'F'   
  AND cust\_email = 'Abbey@company.com'   
UNION ALL   
SELECT \* FROM sh.customers WHERE cust\_gender = 'M'   
  AND cust\_email = 'Abbey@company.com';   
  
Reverse Key Indexes:反转Index key的存储。对于没有反转的index，连续值的index会插入在同一或者相近的block里，造成竞争。而反转的index，连续的index被分 散存储。但是同时，Reverse Key Indexes丢失了range scan的能力。   
  
  
Index Clustering Factor：衡量索引的聚集性，该值越低，则一个范围内的索引更倾向于指向同一个数据块(data block)。   
  
索引的顺序：索引可以定义存储时的顺序，升序或者降序。   
  
Key Compression：索引的Key的存储可以重新排列以节省空间。原有的key可以被分解为prefix entry和suffix entry。   
如   
online,0,AAAPvCAAFAAAAFaAAa   
online,0,AAAPvCAAFAAAAFaAAg   
online,2,AAAPvCAAFAAAAFaAAm   
可以排成   
online,0   
AAAPvCAAFAAAAFaAAa   
AAAPvCAAFAAAAFaAAg   
online,2   
AAAPvCAAFAAAAFaAAm   
  
Bitmap Indexes：对于查询多且取值范围小的列适用。   
Bitmap Join Indexes。   
Bitmap Storage Structure：一样用B-Tree.   
  
Function-Based Indexes:基于函数的索引。   
Application Domain Indexes：基于应用域的索引，用户可以高度定制。   
  
Index Storage：索引可以任意存储，索引和其索引的表不在同一个tablespace中给表备份带来了方便。   
  
Heap-Organized Table   
rows are inserted where they fit。   
  
Index-Organized Tables   
叶子节点直接存储row。以PK作为索引。   
  
两种表的对比   
  
Heap-Organized Table   
The rowid uniquely identifies a row. Primary key constraint may optionally be defined.   
Physical rowid in ROWID pseudocolumn allows building secondary indexes.   
Individual rows may be accessed directly by rowid.   
Sequential full table scan returns all rows in some order.   
Can be stored in a table cluster with other tables.   
Can contain a column of the LONG data type and columns of LOB data types.   
Can contain virtual columns (only relational heap tables are supported).   
  
Index-Organized Table   
Primary key uniquely identifies a row. Primary key constraint must be defined.   
Logical rowid in ROWID pseudocolumn allows building secondary indexes.   
Access to individual rows may be achieved indirectly by primary key.   
A full index scan or fast full index scan returns all rows in some order.   
Cannot be stored in a table cluster.   
Can contain LOB columns but not LONG columns.   
Cannot contain virtual columns.   
  
Index-Organized Tables with Row Overflow Area   
Index-Organized Tables中,为了解决行过大问题，可以把一行分为2部分存储。The index entry储存PK和一些可选列，放在index segment里面，The overflow part包含其他的列，放在storage area segment里面。   
  
Secondary Indexes on Index-Organized Tables   
logical rowids以PK为基础，所以2级索引可以直接构建在logical rowid之上。   
  
Bitmap Indexes on Index-Organized Tables   
  
**4 Partitions, Views, and Other Schema Objects**   
  
Partitioning enables you to decompose very large tables and indexes into smaller and more manageable pieces called partitions. Each partition is an independent object with its own name and optionally its own storage characteristics.   
  
分区的优点   
  
Increased availability   
Easier administration of schema objects   
Reduced contention for shared resources in OLTP systems   
Enhanced query performance in data warehouses   
  
Partition Key   
用来决定分区的键值。   
Partitioning Strategies   
多种分区策略，可以只做一次分区，也可以分区之中再做分区。   
用范围分区，用指定值list分区，用hash分区。   
  
Index一样可以分区：   
local partitioned index：一个索引的分区对应一个表的分区。   
local partitioned index可以分为Local prefixed indexes和Local nonprefixed indexes。   
Local prefixed indexes：The partition keys are on the leading edge of the index definition.   
Local nonprefixed indexes： The partition keys are not on the leading edge of the indexed column list and need not be in the list at all.   
注意查询的时候，可以对Local prefixed indexes做优化，partition elimination。   
  
Global Partitioned Indexes   
索引的分区和表的分区没有了一一对应关系。   
  
Overview of Synonyms   
  
A synonym is an alias for a schema object.   
  
You can create both private and public synonyms. A private synonym is in the schema of a specific user who has control over its availability to others. A public synonym is owned by the user group named PUBLIC and is accessible by every database user.   
  
Synonyms themselves are not securable. When you grant object privileges on a synonym, you are really granting privileges on the underlying object. The synonym is acting only as an alias for the object in the GRANT statement.